Manual de Utilização TcpDump

Universidade Federal de Minas Gerais
Departamento de Ciência da Computação
Laboratório de Software Livre

4 de fevereiro de 2010

Sumário

1 Introdução

O *TcpDump* é uma ferramenta de *sniffer* rodada através da linha de comando, ou seja, é utilizada no monitoramento dos pacotes que trafegam em uma rede de computadores, de modo que ela imprime na saída os cabeçalhos dos pacotes que passam por uma interface de rede definida.

2 Pré-requisitos

Para um correto funcionamento da ferramenta, se faz necessária a instalação do pacote:

- libpcap

3 Instalação

Para instalar o *TcpDump* em uma máquina com o sistema Debian ou Ubuntu, basta seguir os seguintes passo:

- Primeiramente vire o super usuário root:
  ```
  $ su
  ```
- Instale o pacote da ferramenta utilizando o *aptitude*:
  ```
  # aptitude install tcpdump
  ```

Após a conclusão destes passos, o tcpdump já está instalado e pronto para ser utilizado.

4 Monitorando

Nesta parte da documentação, iremos explicar um pouco sobre como utilizar o tcpdump.
OBS.: Todos os comandos daqui pra frente são feitos como usuário root.

4.1 Parâmetros Importantes

Como a utilização da ferramenta se faz via linha de comando, temos que passar alguns parâmetros na chamada do programa, deste modo vamos listar os mais importantes:

Parâmetro	Descrição
-i <interface>	Especificar a interface de rede a ser analisada
-n	Faz com que o TcpDump não resolva nomes ou converta números de portas para seus nomes de serviços
-v	Modo verboso do TcpDump
-w <arquivo>	Grava todo o tráfego em um arquivo estabelecido
-r <arquivo>	Estabelece um arquivo a ser lido, no qual estão especificadas as interfaces a serem analisadas
src host <ip>	Define um ip de origem, no qual a ferramenta só irá analisar os pacotes enviados a partir de um certo endereço ip para a nossa máquina
dst host <ip>	Define um ip de destino, no qual a ferramenta só irá analisar os pacotes enviados da nossa máquina para a que possui o endereço ip declarado
not host <ip>	Define um ip, de modo que toda a rede será analisada, excluindo apenas os pacotes enviados pelo endereço ip declarado
dst port <value>	Define uma porta de conexão, na qual serão analisados os pacotes enviados de nossa máquina através da porta declarada
src port <value>	Define uma porta de conexão, na qual serão analisados os pacotes vindos para nossa máquina através da porta declarada

Outros parâmetros mais avançados podem ser utilizados juntamente com o tcpdump, e uma base melhor sobre eles pode ser compreendida através do manual da ferramenta, acessada com o comando:

```
$ man tcpdump
```

4.2 Exemplos

Para uma maior clareza da utilização do tcpdump, temos alguns exemplos claros com suas respectivas funções:

- Caso vamos analisar todos os dados que passam pela interface eth0, temos o comando:
  ```
  # tcpdump -i eth0
  ```
- Verifica o tráfego destinado à porta 80 através da interface eth0, sem resolver nomes:
  ```
  # tcpdump -i eth0 -n dst port 80
  ```

- Analisa todo o tráfego da rede na interface eth0, mas apenas de conexões do host 192.168.0.1 pela porta 22. Além disto, a saída vai sendo escrita em um arquivo:
  `# tcpdump -i eth0 src host 192.168.0.1 src port 22 -w arquivo`

- Analisa todo o tráfego da primeira interface escrita no arquivo de entrada e imprime a saída em modo verboso:
  `# tcpdump -v -r arquivo`

Por fim, temos uma imagem do tcpdump funcionando. A saída do programa pode não ser muito clara para iniciantes, mas basta conhecer a pilha tcp/ip e os serviços de rede que rapidamente as linhas vão se tornando mais e mais familiares.

```
root@ronaldo:/# tcpdump -i eth0
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
15:00:46.248782 IP climene.speed.dcc.ufmg.br.ipp > 150.164.3.255.ipp: UDP, length 176
15:00:46.249867 IP epiteu.speed.dcc.ufmg.br.ipp > 150.164.3.255.ipp: UDP, length 170
15:00:46.250126 IP ronaldo.speed.dcc.ufmg.br.36414 > aptproxy.speed.dcc.ufmg.br.domain: 10881+ PTR? 255
15:00:46.250513 IP aptproxy.speed.dcc.ufmg.br.domain > ronaldo.speed.dcc.ufmg.br.36414: 10881 NXDomain
15:00:46.351372 IP ronaldo.speed.dcc.ufmg.br.mdns > 224.0.0.251.mdns: 0 PTR (QM)? 255.3.164.150.in-addr
15:00:47.352925 IP ronaldo.speed.dcc.ufmg.br.mdns > 224.0.0.251.mdns: 0 PTR (QM)? 255.3.164.150.in-addr
15:00:47.484546 STP 802.1d, Config, Flags [none], bridge-id 8000.00:1e:58:f0:ac:b0.8007, length 43
15:00:48.421457 IP bs-in-f83.google.com.www > ronaldo.speed.dcc.ufmg.br.35673: P 372905272:372905300(28
0 1219184>
15:00:48.421502 IP ronaldo.speed.dcc.ufmg.br.35673 > bs-in-f83.google.com.www: . ack 28 win 431 <nop,nc
15:00:48.522009 arp who-has router.speed.dcc.ufmg.br tell tanatos.speed.dcc.ufmg.br
15:00:49.316911 IP ronaldo.speed.dcc.ufmg.br > zelda.speed.dcc.ufmg.br: ICMP echo request, id 34831, seq 0, length 30
15:00:49.317156 IP ronaldo.speed.dcc.ufmg.br > topgear.speed.dcc.ufmg.br: ICMP echo request, id 34831, seq 0, length 3
15:00:49.317194 IP zelda.speed.dcc.ufmg.br > ronaldo.speed.dcc.ufmg.br: ICMP echo reply, id 34831, seq 0, length 30
15:00:49.317339 IP topgear.speed.dcc.ufmg.br > ronaldo.speed.dcc.ufmg.br: ICMP echo reply, id 34831, seq 0, length 30
15:00:49.317437 IP ronaldo.speed.dcc.ufmg.br > neverwinter.speed.dcc.ufmg.br: ICMP echo request, id 34831, seq 0, leng
15:00:49.317749 IP neverwinter.speed.dcc.ufmg.br > ronaldo.speed.dcc.ufmg.br: ICMP echo reply, id 34831, seq 0, length
15:00:49.354268 IP ronaldo.speed.dcc.ufmg.br.mdns > 224.0.0.251.mdns: 0 PTR (QM)? 255.3.164.150.in-addr.arpa. (44)
15:00:49.483864 STP 802.1d, Config, Flags [none], bridge-id 8000.00:1e:58:f0:ac:b0.8007, length 43
15:00:51.156334 IP sn1msg1010810.phx.gbl.msnp > ronaldo.speed.dcc.ufmg.br.42142: P 1048317331:1048317421(90) ack 20851
81708 1214638>
15:00:51.156380 IP ronaldo.speed.dcc.ufmg.br.42142 > sn1msg1010810.phx.gbl.msnp: . ack 90 win 59 <nop,nop,timestamp 12
15:00:51.248097 arp who-has aptproxy.speed.dcc.ufmg.br tell ronaldo.speed.dcc.ufmg.br
15:00:51.248408 arp reply aptproxy.speed.dcc.ufmg.br is-at 00:11:11:59:d8:71 (oui Unknown)
15:00:51.252376 IP ronaldo.speed.dcc.ufmg.br.34924 > aptproxy.speed.dcc.ufmg.br.domain: 47611+ PTR? 139.3.164.150.in-a
15:00:51.253107 IP aptproxy.speed.dcc.ufmg.br.domain > ronaldo.speed.dcc.ufmg.br.34924: 47611 2/2/1[|domain]
15:00:51.253281 IP ronaldo.speed.dcc.ufmg.br.42981 > aptproxy.speed.dcc.ufmg.br.domain: 5105+ PTR? 206.3.164.150.in-ad
15:00:51.253779 IP aptproxy.speed.dcc.ufmg.br.domain > ronaldo.speed.dcc.ufmg.br.42981: 5105 2/2/1[|domain]
15:00:51.253898 IP ronaldo.speed.dcc.ufmg.br.56688 > aptproxy.speed.dcc.ufmg.br.domain: 26206+ PTR? 182.3.164.150.in-a
15:00:51.254537 IP aptproxy.speed.dcc.ufmg.br.domain > ronaldo.speed.dcc.ufmg.br.56688: 26206 11/2/1[|domain]
15:00:51.254643 IP ronaldo.speed.dcc.ufmg.br.55241 > aptproxy.speed.dcc.ufmg.br.domain: 62366+ PTR? 202.3.164.150.in-a
15:00:51.255274 IP aptproxy.speed.dcc.ufmg.br.domain > ronaldo.speed.dcc.ufmg.br.55241: 62366 2/2/1[|domain]
```

5 Créditos

Universidade Federal de Minas Gerais
Departamento de Ciência da Computação

João Victor dos Anjos Bárbara
Israel Guerra de Moura
Breno Augusto Vieira Moreira